Episode 8: Understanding the Universe through Molecular Dynamics
Host: Nicole Huesman, Intel
Guests: Erik Lindahl, University of Stockholm; Roland Schulz, Intel

---

Nicole Huesman: Welcome to *Code Together*, an interview series exploring the possibilities of cross-architecture developing with those who live it. I'm your host Nicole Huesman.

Molecular dynamics, and in turn GROMACS, is the bedrock of the Folding@home project. Through molecular dynamics, or MD, scientists and researchers are studying the functional mechanisms of proteins, the process of protein folding, and where drug molecules bind, and how they exert their effects. GROMACS is one of the world's most widely used open source MD applications. Today, I'm happy to welcome two guests to the program who have worked together in this space for many years.

Erik Lindahl is a biophysics professor at Stockholm University and KTH Royal Institute of Technology. And he's been instrumental in the development of GROMACS. Great to have you with us, Erik.

Erik Lindahl: It's great to be on the show, Nicole.

Nicole Huesman: And Roland Schulz, parallel software engineer at Intel. He's worked on GROMACS for over a decade, contributing primarily to parallel efficiency improvements and code modernization. Thanks for joining us, Roland.

Roland Schulz: Thank you for having me.

Nicole Huesman: Erik, can you tell us more about MD and how it benefits the world around us?

Erik Lindahl: So I happen to work on biomolecular systems, but this is something that's used for a whole lot of particle-based systems, everything from the proteins inside our bodies to the largest galaxies in the world. And that has to do with that any objects interacting in the classical world tend to be described by Isaac Newton's equations of motion that all of us probably studied in high school at least. These are approximate equations, but they're approximate in the sense that they tend to describe 99.999% of everything we see around us really well. And that goes for the atoms inside the molecules in our cells too. There are certainly cases when we might need quantum mechanics and other very detailed ways of describing the interaction. But having the detail is not enough. So, I can make a slight analogy that, you know, if you had one of these toys when you were a kid when you were putting blocks of different shapes into small holes, the physicist's way of doing this would be to calculate the equations exactly which box should fit a hole, and then you only had one try to fit the box. But the reason why a two-year-old is much better at that is that the two-year-old simply tests a million times. And that's essentially what we're doing with molecular dynamics too. We're mimicking this way nature has of randomly moving small molecules because temperature gives all atoms a bit of velocity. And the way our molecules will interact and potentially bind each other is that, essentially, they're testing billions of trillions of quadrillions of different conformations in our cells every millisecond.

Now, since we know these interactions, we can describe them reasonably well at least with physics. That means that we can mimic this process inside computers. And this is actually a pretty old setup. It was done first in the 1950s and 1960s with very simple systems. And then even then turned out that we can use physics and computers to understand things that have been virtually impossible to understand from experiments. And particularly we can also understand why they happen.

And then we, as so many other areas of research, have of course benefited tremendously from the semiconductor revolution that Intel very much has been a part of [laughs]. And because computers are now probably a billion times faster than they were in the 1960s, that also means that we can read to scales a billion times longer in time and space. I've actually used this to study things that, even when I was a student, we thought that we would never be able to do in computers. And today, not only do we do them in computers, we're increasingly moving things away from the lab because it's difficult, complicated, wet, and expensive. And to the computer where we can not only do it faster, but also understand why it happens.

Nicole Huesman: What a great overview of MD, Erik. Can you talk about some of the challenges that it brings?

Erik Lindahl: Sure. In essence, it's actually very, very simple. We have one atom here and one atom here. And then we want to calculate how much they're interacting. And that's essentially Coulomb's Law of Interaction. It's a very, very simple calculation. It probably only takes 20 floating point operations or so. The problem is that we have many atoms. We have many, many, many, many atoms. And in addition to that, even when we calculate this how all pairs of atoms in a protein with 100,000 of them interact, we now also need to redo this, because one step only gives us a one-tenth of a second time. So, the problem is not necessarily that it's complicated equations, but that we need to calculate an insane number of times.

That is very simple to solve as long as computers get faster all the time. So, if you only had those one exahertz processors or something, this would be really easy to solve in a single thread. Unfortunately, we're not gonna have exahertz processors. We're gonna need to go parallel instead. And that's where the challenges start to getting these things that should execute in well under a millisecond but do that execution spread over tens of thousands of processors and threads. And that's where people like Roland come in. He's far more skilled at the parallelization aspect of this than I am.

Nicole Huesman: You know, I'm reminded, at International Supercomputing earlier this year, Trish Damkroger talked about the convergence of AI with HPC. Roland, are you seeing this convergence in MD?

Roland Schulz: So, what I definitely see is that the computational challenges Erik mentioned are very similar for both AI as well as MD. The primary hardware requirement is a lot of computational power. That's not actually true for all of HPC. There are quite a few HPC apps

---

which need huge memory or very fast file system and so on. But for MD, it's really focused on, floating point performance your supercomputer has. And that tends to be true for AI too. So as people are pushing the envelope for more and more compute, partly driven by AI, MD benefits too. And then, of course, the technologies we use on the software side with programming languages and so on. And the recent improvement in those areas, there we benefit from MD too.

As far as actually like usage of AI, before you can actually do MD, you need to like have a force field, which describes how all the atoms interact. And the training of those force fields, there I've seen people start using AI technologies. For the actual analysis of MD, I think most people are still sticking to traditional technologies, but Erik would know better than me.

Erik Lindahl: Sure. I'd be happy to expand a bit. AI is occasionally a bit of a buzzword, but the part where this definitely does work is in the analysis. Because when we have these proteins moving over billions of frames, this becomes virtually impossible to analyze with any traditional ways of regression. So general machine learning techniques are used a lot to try to identify what part of a protein is moving. Is there some sort of synchronous motion here that might be correlated to binding or something? And there's been a lot of amazing development in trying to understand what happens with this massive amount of data. And that is definitely used in production, even in all the COVID-19 drug design projects.

Then there is this other part Roland talked about, that there is a limitation to accuracy because we need speed. On the other hand, if you only go for accuracy, we don't get speed, and then we can't handle realistic systems. So, there are trials where people try to use artificial intelligence to predict how atoms would interact. And the nirvana here would be to have the speed with classical MD but the accuracy of quantum chemistry. And there is some very promising development going on there.

The caveat is still that it's still very early. And the problem is not so much that it doesn't work. It does work in some cases. The hard part is predicting when it fails, right? Because I don't think any of us want to use a drug that might have been decided by something that it might have gotten the mechanism wrong. Classical MD is certainly wrong in lots of cases too, like even more cases. But we understand classical MD quite well, and we know what kind of accuracy to expect from it, and we know when it goes wrong, we typically understand why it went wrong.

And both the blessing and curse of AI is that it's occasionally really good, and in other cases it fails miserably. And we don't really understand why it fails, and particularly not when it will fail. But I definitely think we're gonna see more of that in the next few years.

Nicole Huesman: So, Erik, when we talked earlier, you talked about this immense innovation in hardware. And I'd love for you and Roland to talk more about where you see the innovation in hardware going, how you see the evolution in CPUs and accelerators. Where is that headed?

Episode 8: Understanding the Universe through Molecular Dynamics
Host: Nicole Huesman, Intel
Guests: Erik Lindahl, University of Stockholm; Roland Schulz, Intel

---

Erik Lindahl: So in one of the talks I give, I have a plot where I compare computers over the last ... since 1999, I think … and I deliberately have not removed anything. I just keep adding to those plots. On the one hand, this hardware challenge occurs. It's constantly people like Roland and me and my entire team, we keep fighting this, right? And it's a battle of very small wins. We gain effective 10% or maybe 20% if we're really lucky. But I think what's occasionally useful for us is to take a step back and imagine the next generation of machines—whether we call them exascale or post-exascale or something, it's not really that important. But within five or ten years, we're gonna have more floating point operations available than I could ever imagine in my wildest dreams. The price for that though is that you will not have them available in one computer in the sense we think of a computer, one chip or something. But you might have to use, say, a billion functional elements. And whether these functional elements are SIMD units or processor elements in a GPU or something, that doesn't really matter. On the one hand, this is amazing because we will have more power than ever. But the challenge is that it's gonna be spread over so many different elements that no traditional way of parallelizing this is gonna work.

And that kind of brings us some freedom too because it means that you have to completely reinvent. And I think the community is doing that. Like people like Roland, he's had this amazing large-scale simulations with ligno-cellulose, and they've gotten them to scale better than I could ever have imagined. We're doing unsolvables with lots of different molecules and simply get sampling. And it's kind of the same way that a chemistry experiment works. It's very rare to do chemistry experiments with just one molecule. You typically have a billion molecules in your test tube, right?

So, I think that these limitations we're heading to in hardware, that we're not getting those exahertz processors, that's also leading to a lot of innovation. But the innovations on the hardware side that we tend to first see as limitations for us, in the next step that actually leads to innovation in software and science too.

Nicole Huesman: Roland, I'd love your thoughts as well on this topic.

Roland Schulz: I very much agree with Erik that clearly everything's gonna get more parallel, because that's the way of making more powerful machines. We don't really have other ways of doing it. And as Erik said, this makes the challenge of doing those kinds of calculations, which naturally have somewhat limited parallelism, very, very challenging. If you do a global simulation of the Earth, you have automatically a lot of data parallelism, and it might be a bit simpler. But if you only have 100,000 atoms to simulate, spreading those over 100,000 compute units is inherently extremely difficult because every compute unit gets a tiny, tiny amount of work.

So, MD is one of the areas where we see this, people also call it strong scaling, that strong scaling to this very high parallelism is very, very challenging. And I think what we can do from the vendor side is primarily help with software technology tools. So, improving programming languages and other tools, given that scientists will need to do more and more

to enable these computers. At least the tools should be as good as possible to make the work as painless as possible.

Nicole Huesman: Absolutely. If you were able to design either a solution stack or tools tailored specifically for MD, what features or capabilities would this stack or these tools have?

Erik Lindahl: I don't want the stack specific for MD. And that might sound very strange, but science has tried this. So, the problem is that it's not difficult to make even a hardware that's specific to something. It's been tried in bioinformatics. It's gonna cost you $100 million or something, and then the vendor will sell three copies of it, and then they will close business a year later, and then all that effort that we invested in porting for that hardware will be lost.

And we've tried this over and over again. And I would argue that the reason why scientists in general are so happy with a Linux, x86, and the standard solutions is that it keeps paying back, right? Because we are investing in something that the entire community ... And now I'm not even thinking scientific community, but the worldwide community in computing … we develop it together. So, if I got something that was tailor made for me, I would also need to have a business and market that could sustain the development. But I don't think that's possible for any branch of science.

So surprisingly, I like the straight jacket of having languages such as CUDA or SYCL that tell me like, "Look, it's not magic. I can't just write my signs and then have the compiler magically translate it." That would be really nice, but it doesn't work in practice. So, what I like with these parallel languages is that without going into all the detail about the hardware, they tell me that, "Erik, to express the fundamental parallelism in your scientific problem in the algorithm, that's my job." And I have to do that as a scientist, and we're reasonably good at that, because I understand my algorithm well, and I even think I understand my algorithm better than the compiler does. So, I am able to identify that parallelism. Then, of course, all the bolts, all the details, the way to translate this into different hardware generations, I have no idea about that. The compiler's much better at that. But I think that's the genius, and I think one should give NVIDIA credit here because CUDA was the first really good language at doing this. And that's ... What I like about SYCL in particular now is that we're getting portable languages to do this that I as a scientist have to express the parallelism in the algorithm, and then I have tools that can help me with everything else.

Nicole Huesman: So, let's shift a little bit and talk about GROMACS. Erik, can you talk a little bit about how important GROMACS software is in the fight against COVID, and also what it means to you to have been such a big part of its development?

Erik Lindahl: I think COVID is a very difficult situation in the world, and I don't think any single technology is gonna solve this, with the possible of some parts of vaccine development, but even vaccine development is probably a dozen different techniques

internally. And I certainly don't think molecular simulation on its own is gonna solve this. What I'm seeing in the world, and this is not just the consortium we are involved in, but several users, is that MD is helping us to understand, for instance, how the different parts of a dimer are moving together, when a ligand is binding to some of these proteins, how does it change the function. So I think at this stage, MD is primarily helping us with understanding how things work on the molecular level and possibly helping us to find candidates that are worth testing. At this stage, I think all of the real tests will still have to be done in the lab to confirm things.

On the other hand, had you asked me 10 years ago, I never thought that we would have been able to do even this much. So give this 10 or 20 years, and I might actually have to eat some of that stuff I'm saying because at that point, we might do a whole lot more of the actual antiviral design in computers. As scientists, on the one hand, we love our science. And I love these equations of motions. And I love just sitting with code and getting computers to do things that I thought that they couldn't do [laughs]. But there's also a deep satisfaction that what you're doing is productive and helping other people. And I think in the purest sense of egotism, right, that we want meaning to what we're doing. And particularly if you're working long hours, that it matters is important. Not necessarily in terms of money or anything, but those hours are not wasted.

And in science too there are ups and downs. But in particular in those downs and when you're swearing a bit, you realize, well, at least on average there were 10 more papers published today where people used whatever you were using. And that gives a bit of long-term satisfaction. Not just a bit actually. I think it matters a lot if you're helping other people advance science.

Nicole Huesman: Roland, we'd also love for you to chime in on this. You've been instrumental as well in the advancement of GROMACS. In your view, what have been some of the most promising developments?

Roland Schulz: So from a computational perspective, the biggest improvements we've done are, on the one hand, scaling to very many nodes. GROMACS made a lot of improvements in that area over the years. Berk Hess has done amazing work with like really good domain decomposition methods, minimizing the amount of communication needed between them and then coupling this with the PME method for the long-range electrostatic where it helped us to scale that part also to many nodes. So we've done, yeah, a lot of work to go to many nodes.

And then of course on the compute side within the node, on the one hand, you can use accelerators, and the improvements with accelerators over the last few years has been very strong, and GROMACS can make good use of those. On the other hand, there are still lots of clusters out there which also use CPUs, and we've made a lot of improvements towards making best use of the SIMD units. GROMACS has a custom SIMD library to make really efficient use of the SIMD units within CPUs.

Episode 8: Understanding the Universe through Molecular Dynamics
Host: Nicole Huesman, Intel
Guests: Erik Lindahl, University of Stockholm; Roland Schulz, Intel

---

Nicole Huesman: As one of those who is actually currently running Folding@home on my iMac, I was really excited to see the project surpass the exascale barrier, exceeding the compute performance of the top 500 supercomputers. And I wonder, Roland, how do you see Folding and GROMACS, you know, one of the engines that's powering Folding, how do you see these growing in the future?

Roland Schulz: It's amazing that people can contribute their CPU time they're not using themselves towards helping science. The comparison to supercomputers, one has to be little bit careful with. There are really amazing things one can do with distributed computing projects like Folding@home. But they are also somewhat limited in what type of calculations you can do. Erik earlier mentioned the ensemble method. So if you have very many small molecules, you can give each small molecule to one of the volunteers and they can independently calculate it. And then this works really great. If you have larger proteins, that becomes difficult because what supercomputers have, they have this really fast network so we can parallelize the work over more than one node. The internet isn't fast enough for that. So for these distributed computing projects, we can't do these large molecules and distribute the work across multiple ones of those. Those still have to run on real supercomputers. But there's, yeah, a lot of research which we now can move towards distributed computing to keep the supercomputer for those works where they're really absolutely necessary.

Erik Lindahl: I came up with a bit of an analogy there. So we all started out doing this the supercomputer way, and we wanted the largest possible computers, which when I was a student was roughly 40 CPUs or something, which is what I have in my desktop nowadays [laughs]. And I guess an analogy there … Assume that you would like to map out the public transit system in New York. The way we historically did things that we would hire one volunteer and pay this person to try to map out the entire public transit system. That would take awhile. While you could argue that the way Folding@home does it is that they get 100,000 people and track them all with their cellphones. Now, that's of course amazing, right? But that does not mean that you could take 10 billion people, because everyone in the world is not in New York City first.

And the other part, if the time scale I can cover is so short that each person only has time to take one step, that's not gonna help me to map out the public transit system in New York. The tracking time has to be at least long enough to cover part of their actual trip. Otherwise, it's not gonna help. And this is where we need supercomputers, too, because for some things, say, a protein on the surface of COVID binding a ligand or something, that can take a microsecond, which is a very short time in the lab, but it's a very long time in the simulations. So there are lots of processes where the individual desktop computers are not fast enough, and for that we do need the supercomputers.

Nicole Huesman: So, Erik, what does this mean for exascale?

Erik Lindahl: I think we're mistaken. We're seeing exascale as a goal. It's not. It's a milestone, right, that we're gonna fly by it, I hope. So at one point in time when I was a student, we had

---

these amazingly fast computers. They were creating machines that like ... this was probably around 1998 … and we had computers within the ballpark of, say, probably 100 teraflop or so. But that type of computing we have in our pockets today, in our cellphones. And, of course, when that first computer was built, people probably said, "Do we even need this amount of computing power?" And we've kept asking, "Do we even need this amount of computing power?"

But what we today call exascale, just wait 20 years [laughs], I bet you're gonna have it in your pocket. And this is gonna have our personal assistant do things that we can't even imagine today. So this is a process that will keep being a challenge, but it will also keep giving back. So the point is not to build the exascale machine and stop. It's the next milestone.

Nicole Huesman: Great point. So as we wrap up today, Erik, can you tell our listeners where they can go for more information about developments in MD and GROMACS?

Erik Lindahl: Sure. Anything is available at http://www.gromacs.org/. That site is a typical scientific site that can be a bit boring and everything. But the cool thing that this entire project is open. So you can track everything that's happening in development in GitLab and everything. There are mailing lists. And quite a few of those are active on Twitter too in these discussions.

If you want a great general introduction to MD, I would actually very much recommend the Folding@home sites. They're using several tools, not just GROMACS. But there are a bunch of beautiful blog posts and video explanations explaining science in general, molecular dynamics in particular, and molecular dynamics of protein folding in very much particular. And I think it's a great resource to get started.

Nicole Huesman: And, Roland, is there anything more you'd like to share with folks? Any other resources they should check out?

Roland Schulz: Yes, the same as for GROMACS, all the SYCL is in open too. So both the SYCL development we do specifically for GROMACS, that is available on the GROMACS GitLab. And then all the SYCL development from the SYCL compiler is on GitHub. And so both of those are in open post, development of the planning as well as the actual source.

Nicole Huesman: Great. Thank you. And there's good news here, and something that I'm really excited about. We're gonna be picking up this conversation next month with Erik and Roland where we'll talk about porting GROMACS across heterogeneous architectures. So, listeners, if you have any questions for Erik and Roland, drop them into the Twitter feed @IntelDevTools, and we'll work them into the next discussion. We touched lightly on CUDA and SYCL, and we're gonna talk a lot more about the tools available to you in our next discussion. So we invite you to tune into that one as well.

For now, Erik, thanks so much for your insights. This work is so vital in addressing our global pandemic.

Episode 8: Understanding the Universe through Molecular Dynamics
Host: Nicole Huesman, Intel
Guests: Erik Lindahl, University of Stockholm; Roland Schulz, Intel

---

Erik Lindahl: Thank you so much, Nicole. It was a great first discussion. And I'm looking forward to going into the gory details next time.

Nicole Huesman: Excellent. And Roland, your knowledge is so invaluable. Thanks so much for being with us.

Roland Schulz: Thank you.

Nicole Huesman: And thanks to all of you out there for joining us. Until next time, thanks for listening.